# Research on Scheduling Algorithm of Cloud Computing Task

**LI Mei-an*, ZHANG Pei-qiang, WANG Bu-yu**

*College of computer and information engineering, Inner Mogolia agriculture university, Huhehaote,Inner Mongolia, 010018,China*

Abstract

For ensuring the computation time of a set of tasks is the shortest, or other scheduling goals, heuristic or other scheduling algorithms had been proposed to schedule the cloud task. But those scheduling algorithms all only resolve one aspect scheduling goal and introduce some other drawbacks such as the scheduling time is too long. For resolving two or more aspects goals and reduce the scheduling time for cloud computing scheduling at the same time, a new task scheduling algorithm based on single ant colony algorithm and Min-Min algorithm(SCMM) for scheduling cloud task. This algorithm cluster the cloud task at first based on the single ant colony algorithm, then schedule the task cluster to the computation nodes based on Min-Min algorithm. Experiment shows that this algorithm(SCMM) can ensure the shortest executing time and higher load balance capability for cloud computing system and will not increase the scheduling time significantly.

Key words: TASK SCHEDULING ALGORITHM, CLOUD COMPUTING, SINGLE ANT COLONY ALGORITHM, MIN-MIN ALGORITHM

## 1. Introduction

Most of the current task scheduling algorithms focuses on solving one aspect problem of cloud computing system. For ensuring the computation time of a set of tasks is the shortest, the scheduling algorithms based on heuristic algorithm[1-4] such as ant colony algorithm(ACA) had been proposed. These algorithms make the task scheduling algorithm have a certain adaptive capacity and improve the overall performance of the cloud computing system. But there is also a higher load of scheduling process and a longer scheduling time. At the same time, cloud task scheduling strategy based on trust model introduces the concept of trust degree to relate resources and tasks[5-8]. In this algorithm, the users not only pay attention to the time cost of the task execution, but also the economic cost of the resource use. Some scholars proposed a scheduling strategy to guarantee the quality of user services [9-15]. In this algorithm, People can preprocess the tasks according to the incidental QoS parameter when user submit his task and search for exact resources to execute this task.

Summary of these current cloud task scheduling algorithms, found that even those scheduling algorithm basing on QoS, in essence, can not meet the needs of the system and the user two or more goals at the same time.

In order to meet the requirements of two or more goals, and ensure the running time of the algorithm is not significantly increased, the single ant colony clustering and Min-Min scheduling algorithm (SCMM) is proposed in this paper. This algorithm is not directly using ant colony algorithm for task scheduling, but the task of clustering. On this basis, using Min-Min algorithm to schedule task clusters, not only reduces the scheduling algorithm running time, but also to maintain the system load balance, and ensure the shortest task execution time.

## 2. Algorithm principle description

### 2.1. Single ant colony clustering cloud task

Single ant colony algorithm classes the cloud task objects distributed in the data plane randomly to some simple classifications, and use some virtual ant colonies with different speed to imitate multi ant colonies to cluster analysis for cloud task object.

**A. Average similarity**

Ants select cloud task object randomly at the initial time, calculate the similarity in the neighborhood of this task, decide to do which operation including lift, transfer and discard to those tasks. Computation of the operating probability of cloud task objects as formula (1)

$$p(o_i) = \max\left\{0, \frac{1}{s^2}\sum_{\delta \in aoc_{s\times s}(r)}\left[1 - \frac{dis(o_i, o_j)}{\alpha(1+\frac{v-1}{v_{\max}})}\right]\right\} \quad (1)$$

In formula(1), $\alpha$ is the similarity empirical parameters. $v$ is the move speed of ant. The more $v$ the cloud task object classification is faster. $aoc_{s\times s}(r)$ is the circular region with $r$ as the center, $s$ as the radius. $dis(o_i, o_j)$ is the distance in attribute space of object $o_i$ and $o_j$. Using cosine distance function, formula (2) and (3) will be gotten.

$$dis(o_i, o_j) = 1 - sim(o_i, o_j) \quad (2)$$

$$sim(o_i, o_j) = \frac{\sum_{k=1}^{n}(o_{ix}, o_{jx})}{\sqrt{\sum_{k=1}^{n}(o_{ix})^2 \cdot \sum_{k=1}^{n}(o_{jx})^2}} \quad (3)$$

$sim(o_i, o_j)$ is the cosine similarity function of the angle of the two task vector in the plane. The greater the similarity , $sim(o_i, o_j)$ is close to 1.

**B. Operation probability transformation**

The similarity between the task objects is represented by the probability transformation function $p_x$. The smaller the average similarity between the task object $o_i$ and the objects in the neighborhood, the lower the probability that the task belongs to the neighborhood, and the greater probability of ants choosing lift operation to deal with the task. Definition the probability of zero load ant(Load identification is 0) lift the cloud task object as formula(4).

$$p_p = 1 - Sigmoid(p(o_i)) \quad (4)$$

Definition the probability of lord ant(Load identification is 1) discard the cloud task object as formula(5)

$$p_d = Sigmoid(p(o_i)) \quad (5)$$

$$Sigmoid = \frac{1 - e^{cx}}{1 + e^{cx}} \quad (6)$$

In formula(6), $c$ is an empirical parameter for the convergence of the algorithm. Simulation results show that the greater $c$, the more easy converge of the algorithm.

**C. Using the ant colony algorithm principle to transfer cloud tasks**

When ants do "transfer" to the cloud task objects, the task object will be transferred to the corresponding task according the principle of ant colony algorithm. Use $G = (V, E)$ to describe $n$ independent tasks on data center, in which $V$ denotes the set of task attribute, $E$ denotes the attributes of every task. Propose there are $M$ ants, and each ant has its own exclusive list used to store those tasks which have been sorted out. Use $Allowed = \{V - Tabu\}$ to describe the set of tasks which have not been sort out. $c_{ij}(t)$ denotes the expect factor that ant transfer task $i$ to task class $j$ at this time. It's value will be decided by the similarity of task $i$ and all tasks in task class $j$. So $c_{ij}(t) = 1/p(o_j)$. $\eta_{ij}(t)$ denotes the pheromone quantity on the road of task class $j$ at $t$ moment. At the initial moment, let $\eta_{ij}(t) = C$ ($C$ is constant). $p_{ij}^{k}(t)$ denotes the transfer probability. Define it as formula(7).

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[c_{ij}(t)]^A * [\eta_{ij}(t)]^B}{\sum_{s \in Allowed_k}[c_{ij}(t)]^A * [\eta_{ij}(t)]^B} & j \in Allowed_k \\ 0 & otherwise \end{cases} \quad (7)$$

In formula(7) $A$ is the expected weight coefficient, and it represents the similarity effect. $B$ is the weight coefficient of pheromone, and it reflects the effect of pheromone on the movement of ants. Ants complete once transfer, and update the pheromone on the $path(i, j)$ according formula(8).

$$\eta_{ij}(t+n) = (1-Q)\eta_{ij}(t) + \Delta\eta_{ij} \quad (8)$$

$$\Delta\eta_{ij} = \sum_{k=1}^{m}\Delta\eta_{ij}^{k} \quad (9)$$

In formula(8), $Q$ is volatile coefficient, and $Q \in (0,1)$, $\Delta\eta_{ij}$ is the pheromone increment leaved in this transfer on $path(i, j)$. $\Delta\eta_{ij}^{k}$ is the pheromone increment in this iteration on $path(i, j)$. Calculate it as formula(10).

$$\Delta\eta_{ij}^{k} = \frac{W}{L_k} \quad (10)$$

In formula (10), $W$ is pheromone intensity, $L_k$ is total cost of this iteration.

**2.2 Min-Min task scheduling strategy**

Use SCMM algorithm to schedule cloud task, need cluster the cloud tasks at first, then store the task set after clustering in a large set $S$, and mark those task sets increasing number. Schedule the task set by serial number. Describe the scheduling process as follow:

When $S!=\varnothing$, take the first element in $S$, execute the follow operation:

Calculate the earliest completion time matrix $E_{CT}$. Select any column of $E_{CT}$, like $E_{CT}(*,j)$, use the maximum value of $E_{CT}(*,j)$ to divide all elements of $E_{CT}(*,j)$, a weight vector reflected the task scale $T_{SW}$ will be gotten. The minimum value $SW_{*J}$ in $T_{SW}$ is corresponding the minimum task scale.

$$T_{SW}=\{SW_{*1},SW_{*2},,,SW_{*n}\} \quad (11)$$

Similarly, Select any row of $E_{CT}$ like $E_{CT}(i,*)$, use the maximum value of $E_{CT}(i,*)$ to divide all elements of $E_{CT}(i,*)$, a weight vector reflected resource performance $S_{IW}$. The minimum value in $S_{IW}$ is corresponding the resource with the fastest execution speed.

$$S_{IW}=\{IW_{1*},IW_{2*},,,IW_{n*}\} \quad (12)$$

Definite the load balancing matrix LBM (Load Balancing Matrix) as follow:

$$LBM=T_{SW}(i,j)\times S_{IW}(i,j)\times E_{CT}(i,j) \quad (13)$$

Call Min-Min algorithm to schedule the task in LBM array until all task had been scheduled, then delete the task set in $S$.

**3. Cloud task scheduling model**

Experiments using two stage scheduling model for task and resource scheduling to three layer service architecture of cloud system. The first stage assigns the task to a virtual resource. The second stage searches the suitable physical resource for virtual machine scheduling according to the task scale and the work situation of the virtual nodes.

**Definition 1: Task unit** denotes the independent task in ready queue. Measure the task unit by three indicators including calculation length, transmission length and storage length.

**Definition 2: Available resource** denotes these virtual nodes ready to schedule in cloud calculating environment. Use set $P$ to store the resource nodes and their performance data including calculation performance, transmission performance and storage performance.

**Definition 3:** $E_{TC}(i,j)$ denotes the predictive execution time of task $i$ on resource $j$. $i\in\{1,2,,,n\}$, $j\in\{1,2,,,m\}$, $i$, $j$ can be calculated by the calculation length of task $i$ and the calculation performance of resource $j$.

**Definition 4:** $E_{TS}(i,j)$ denotes the predictive storage time of task $i$ on resource $j$. $i\in\{1,2,,,n\}$, $j\in\{1,2,,,m\}$, $i$, $j$ can be calculated by the storage length of task $i$ and the storage performance of resource $j$.

**Definition 5:** $E_{TD}(i,j)$ denotes the predictive transmission time of task $i$ on resource $j$. $i\in\{1,2,,,n\}$, $j\in\{1,2,,,m\}$, $i$, $j$ can be calculated by the transmission length of task $i$ and the transmission performance of resource $j$.

**Definition 6:** $E_{CT}(i,j)$ denotes the earliest finish time of task $i$ on resource $j$. Calculate it as formula(14).

$$E_{CT}(i,j)=E_{TC}(i,j)+E_{TS}(i,j)+E_{DT}(i,j) \quad (14)$$

Calculate the $E_{CT}(n,m)$ array by the execution time, transmission time and storage time of task $i$ on resource $j$.

**4. Algorithm description**

Describe our SCMM algorithm as follow.

```
Begin
    Init(s,,c,A,B,G,R,M,vmax,…);// Initialize the parameters of system.
    Projection(JobList){ // Project the task to the data plane, and calculate the coordinates (x, y).
    For(int i=0;i<M;i++){
    Set(ant=Ji,ant.flag=1);// Let M ants randomly select the task objects and set the ant load mark as 1;
    Init(Tabu);// Initialize ant Tabu list;
    For(int j=n;j>n;j--){// Clustering for each task. Calculate the average similarity of object I and the tasks in neighborhood P(oi) according formula(7). Calculate Pp, Pd according formula(4),formula(5).
    If(ant.flag==0 && Pp>0.5){
     Pickup(Ji);//No load ants lift task i if the average similarity of task i and task set j.
     Calculate and transfer task I to task set j;
  Update(pheromenon);// Update pheromone;
     Update(ant.flag); // Update load identification;
    }Else If(ant.flag==1 && Pd>0.5){
          Putdown(Ji); // Load ants discard task i.
          Update(ant.flag);
          }Else{
          Special(Ji);// Set task I isolated, and process it later }}}
  For(k=1;k<N;k++){ //N is the task number in task set S;
   While(S!=){
```

ECT(i,j)=ETC(i,j)+ETS(i,j)+ETD(i,j);// Predict the earliest completion time of the task set S;

LBM=TSW(i,j)×(SIW(i,j)×ECT(i,j));//Calculate the load balancing matrix LBM of the task set S;

While(task != ){

Mintime[]=min(LBM[][]);//Take the minimum value of each column of the LBM matrix to find the machine with the best performance;

Mintime=min(Mintime[]);

CloudTask.hasNext()=i;//Let Mintime to be corresponding task i. Assign task i to the execution queue.

CloudResourse.hasNext()=j;//Let Mintime to be corresponding physical resource j. schedule physical resourceh j.

Delete(task i); }

Delete(task set k);}}}

End;

## 5. Experimental and performance analysis

### 5.1. Experimental deployment

The scheduling model of SCMM is similar to dynamic adaptive ant colony algorithm(DAACA) [15]. They all are aimed at the goal of time performance and Resource utilization rate. In this experiment the experimental environment will adopt the same as the DAACA and adopt the datacenter.txt as our task set. Show the initial algorithm parameter as table 1. In which, $s$ represents the clustering radius, $\alpha$ is the Empirical parameter of similarity, $c$ is the Empirical parameter of adjusting the convergence speed of SCMM. $v_{\max}$ is the maximum speed limited by SCMM of ant. $A$, $B$ are the weight factor of heuristic information and pheromone concentration in SCMM. $C$ is initial pheromone residual value, $Q$ is the pheromone evaporation coefficient and $1-Q$ is the pheromone residual coefficient. $Num_{\max}$ is the maximum iteration times of SCMM and M is the number of ants.

**Table. 1.** Initialization parameter table of SCMM

| Name | s | α | c | $V_{max}$ | A | B | C | Q | $Num_{ma}$ | M |
|---|---|---|---|---|---|---|---|---|---|---|
| Value | 5 | 0.6 | 2 | 5 | 1 | 2 | 3 | 2 | 500 | 50 |

### 5.2. Performance analysis

In this experiment, select the first ten tasks in datacenter.txt as the task set. Then simulate the algorithms of SCMM and DAACA through this experiment. Figure 1 shows the comparison of implement times of SCMM and DAACA for solving the scheduling problems of the first 10 tasks in datacenter.txt.

From figure 1, the time span of DAACA is 78.69 seconds, and the time span of SCAA is 79.10 seconds for solving the scheduling problems of the first 10 tasks in datacenter.txt. That's to say, there is no obvious difference of time cost of these two algorithms when the task scale is smaller. Figure 4 shows the complete task number of every node.

Simulate the algorithms of SCMM, DAACA and colony algorithm(ACA) through experiment. Figure 2 shows the execute time of SCMM, DAACA and ACA when the task set is datacenter.txt.

Definition the load balance capability(LBC) as follow:

$$Load=\sqrt{\frac{1}{n}\sum_{i=1}^{m}(this.MI-systhis.MI)^2}$$

In this formula, this.MI represent the load in resource $i$, systhis.MI is the average load of every node in system. Figure 3 shows the LBC of SCMM, DAACA and ACA for solving the scheduling problem of all 400 tasks in task set datacenter.txt and propose there are eight virtual machines in this experiment. From figure 3, we know that SCMM improve the load balance capability 27.6% than DAACA.

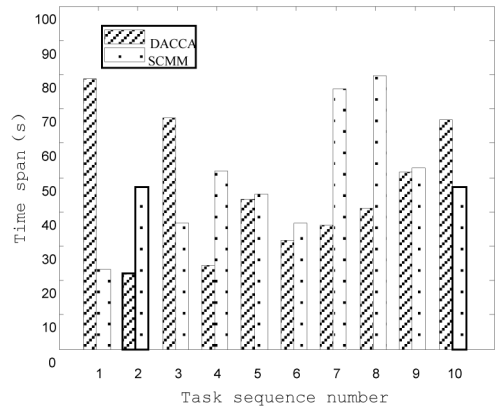


**Figure 1.** The comparison of task scheduling time

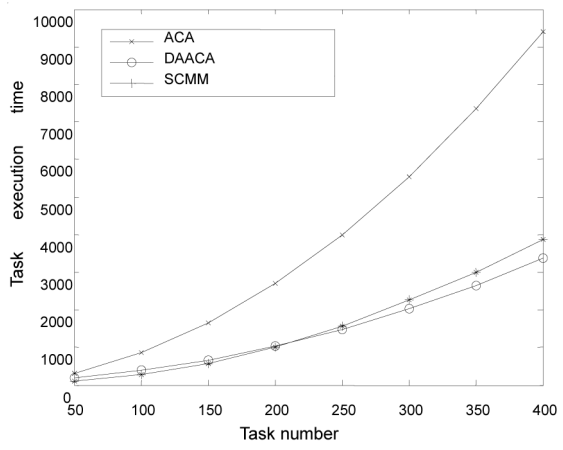


**Figure 2.** The time span of three algorithms

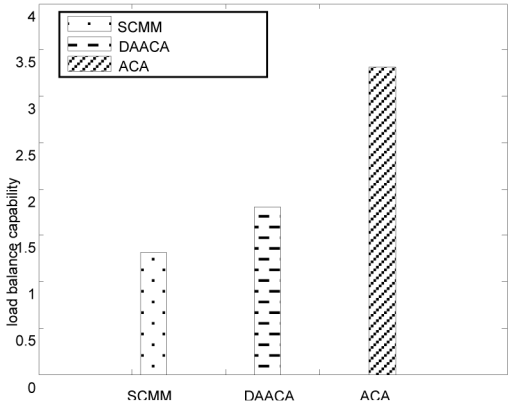


**Figure 3.** Load factor of three algorithms

**6. Acknowledgments**

This study is supported by the building project of Inner Mongolia Agricultural University science and technology innovation team(NDPYDT2010-9) .

**7. Conclusion**

SCMM proposed in this paper can improve the load balance capability than DAACA and ACA significantly when there is the same task set. And the same scheduling time of SCMM with DAACA when there is the same task set. So SCMM improves the resource utilization ratio of the cloud system and ensure the schedule times of two or more scheduling algorithm are same.

**References**

1. Elina Pacini, Cristian Mateos, Carlos García Garino, Balancing throughput and response time in online scientific Clouds via Ant Colony Optimization. Advances in Engineering Software, 2015,84,p.p.31-47.
2. Zhang Yan-hua, Feng Lei, Yang Zhi, Optimization of Cloud Database Route Scheduling Based on Combination of Genetic Algorithm and Ant Colony Algorithm, Procedia Engineering, 2011,15, p.p.3341-3345.
3. Mala Kalra, Sarbjeet Singh, A review of metaheuristic scheduling techniques in cloud computing, Egyptian Informatics Journal, Available online 18 August 2015, ISSN 1110-8665.
4. Sucha Smanchat, Kanchana Viriyapant, Taxonomies of workflow scheduling problem and techniques in the cloud, Future Generation Computer Systems, 2015,52,p.p.1-12.
5. Tom Guérout, Samir Medjiah, Georges Da Costa, Thierry Monteil, Quality of service modeling for green scheduling in Clouds, Sustainable Computing: Informatics and Systems, 2014,4,p.p.225-240.
6. Rahul Urgaonkar, Shiqiang Wang, Ting He, Murtaza Zafer, Kevin Chan, Kin K. Leung, Dynamic service migration and workload scheduling in edge-clouds, Performance Evaluation, 2015,91,p.p. 205-228.
7. Youwei Ding, Xiaolin Qin, Liang Liu, Taochun Wang, Energy efficient scheduling of virtual machines in cloud with deadline constraint, Future Generation Computer Systems, 2015,50,p.p.62-74.
8. Mohammed Abdullahi, Md Asri Ngadi, Shafi'i Muhammad Abdulhamid, Symbiotic Organism Search optimization based task scheduling in cloud computing environment, Future Generation Computer Systems, Available online 24,2015.
9. Igor Bilogrevic, Murtuza Jadliwala, Praveen Kumar, Sudeep Singh Walia, Jean-Pierre Hubaux, Imad Aad, Valtteri Niemi, Meetings through the cloud: Privacy-preserving scheduling on mobile devices, Journal of Systems and Software, 2011,11,p.p.1910-1927.
10. Xiaomin Zhu, Rong Ge, Jinguang Sun, Chuan He, 3E: Energy-efficient elastic scheduling for independent tasks in heterogeneous computing systems, Journal of Systems and Software, 2013, 2,p.p.302-314.
11. Mads Darø Kristensen, Niels Olof Bouvin, Scheduling and development support in the Scavenger cyber foraging system, Pervasive and Mobile Computing, 2010, 6,p.p.677-692.
12. Mye Sohn, Sunghwan Jeong, Jongmo Kim, Hyun Jung Lee, Augmented context-based recommendation service framework using knowledge over the Linked Open Data cloud, Pervasive and Mobile Computing, Available online 22,2015.
13. Chengying Mao, Jifu Chen, Dave Towey, Jinfu Chen, Xiaoyuan Xie, Search-based QoS ranking prediction for web services in cloud environments, Future Generation Computer Systems, 2015, 50,p.p111-126.
14. Shih-Chun Lin, Pu Wang, Min Luo, Jointly optimized QoS-aware virtualization and routing in software defined networks, Computer Networks, Available online 2, 2015.
15. WANG Fang, LI Mei-an, DUAN Wei-jun. Cloud computing task scheduling based on dynamically adaptive ant colony algorithm,Journal of Computer Applications. 2013,11,p.p.3160-3162.